### **ASSIGNMENT 4**

Student Name	Balaji.V
Student Roll Number	820519205009
Maximum Marks	2 Marks
Project Name	Containment Zone Alerting Application
Team ID	PNT2022TMID52453

## **Questions:**

- 1. Pull an image from docker hub and run it in docker playground.
- 2. Create a docker file for the jobportal application and deploy it in Docker desktop application.
- 3. Create a docker file for the jobportal application and deploy it in Docker desktop application.
- 4. Create a IBM container registry and deploy helloworld app or jobportal app.
- 5. Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run in nodeport

## **Output:**

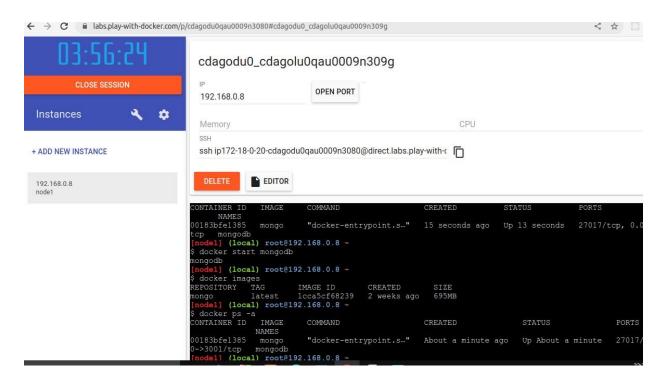
## Pull image in docker:

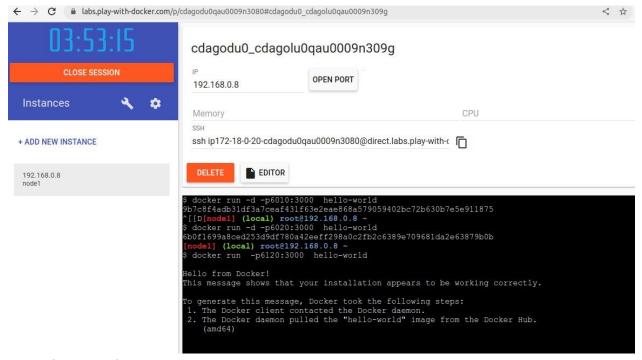
```
13-alpine3.10
10.10
                                                               9e3fc25f69d2
9a05a2b9e69f
                                                                                                         114MB
211MB
The push refers to repository [icr.io/helloind/python] f7a508b1c151: Pushed
7216ce2329ba: Pushed
of61ccc2347a:
ofc1deb8136e:
                  Pushed
Lf123186824c:
3d6eb1152931:
                  Pushed
Pushing
                                                                                                         7.748MB/57.12MB
100796cdf3b1:
54acb5a6fa0b:
                                                                                                         11.14MB/18.48MB
4.294MB/528.7MB
                  Pushing
                  Pushing
3d51c618126f:
9ff6e4d46744:
                  Pushing
Waiting
                                                                                                         1.063MB/152MB
89d1d47b5al: Waiting
555ed1b7a428: Waiting
'Croot@admir/home/ash/Desktop/Flask ClassWorrk# docker push icr.io/helloind/python:1.0
The push refers to repository [icr.io/helloind/python]
Ta588blc151: Layer already exists
77b353df37a8: Layer already exists
7216ce2329ba: Layer already exists
9761cc2329ba: Layer already exists
ofcldeb8136e: Layer already exists
Lf123186824c: Layer already exists
3d6eb1152931: Pushed
100796cdf3b1:
                  Pushed
54acb5a6fa0b:
                  Pushed
3d51c618126f:
)ff6e4d46744:
                  Pushed
a89d1d47b5a1:
                   Pushed
555ed1b7a428:
L.O: digest: sha256:9d7b9f369391c18d6aa88270ebb467ab6b93b3f18e61a46c9467b7fa5cc4b404 size: 3050
```

#### **Run Docker:**

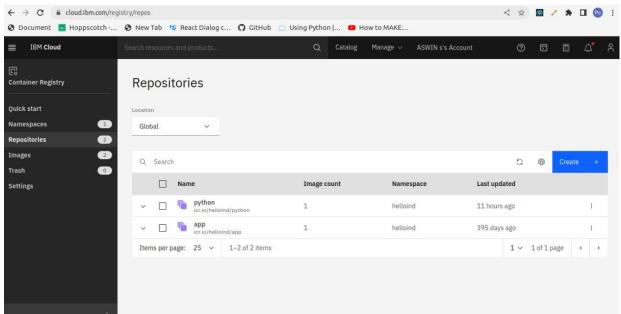
```
Login Succeeded
Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/ac
root@admin:/home/ash/Desktop/Flask ClassWorrk# docker run -p5000:5000 flakapp:1.0
Unable to find image 'flakapp:1.0' locally
 docker: Error response from daemon: pull access denied for flakapp, repository does not exist or may require 'docker lo
quested access to the resource is denied.
See 'docker run --help'.
 root@admin:/home/ash/Desktop/Flask ClassWorrk# docker run -p5000:5000 -t flakapp:1.0
Unable to find image 'flakapp:1.0' locally
 docker: Error response from daemon: pull access denied for flakapp, repository does not exist or may require 'docker lo
quested access to the resource is denied.
See 'docker run --help'.
 root@admin:/home/ash/Desktop/Flask ClassWorrk# docker run -p5000:5000 -t flakapp
 Unable to find image 'flakapp:latest' locally
 docker: Error response from daemon: pull access denied for flakapp, repository does not exist or may require 'docker lo
 quested access to the resource is denied.
See 'docker run --help'.
 root@admin:/home/ash/Desktop/Flask ClassWorrk# docker run -p5000:5000 -t flaskapp:1.0
 docker: Error response from daemon: driver failed programming external connectivity on endpoint silly_bardeen (717385e0 d59728f1e3577afe801b18edb0257c07b377889d): Bind for 0.0.0.0:5000 failed: port is already allocated.
 ERRO[0002] error waiting for container: context canceled
root@admin:/home/ash/Desktop/Flask ClassWorrk# docker run
                                                                               -t flaskapp:1.0
    Serving Flask app 'app
 * Debug mode: off
 /ARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
  * Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
  * Running on http://172.17.0.3:5000
172.17.0.1 - - [23/0ct/2022 17:21:29] "GET / HTTP/1.1" 200 - 172.17.0.1 - - [23/0ct/2022 17:21:29] "GET / favicon.ico HTTP. 172.17.0.1 - - [23/0ct/2022 17:21:50] "GET / HTTP/1.1" 200 -
```

### **Docker Playground:**





# **Container Registry:**



#### Templates was Created successfully Welcomee

ha

## **KUBERNETES:**

## dashboard-adminuser.yaml

apiVersion: v1

kind: ServiceAccount

metadata:

name: admin-user

namespace: kubernetes-dashboard

---

apiVersion: v1 kind: Secret metadata:

name: admin-user-token

namespace: kubernetes-dashboard

annotations:

kubernetes.io/service-account.name: admin-user

type: kubernetes.io/service-account-token

---

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRoleBinding

metadata:

name: admin-user

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole name: cluster-admin

subjects:

- kind: ServiceAccount name: admin-user

```
namespace: kubernetes-dashboard
Flask-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
name: flask-app
spec:
 replicas: 3
selector:
 matchLabels:
   app: flask-app
template:
  metadata:
   labels:
    app: flask-app
  spec:
   containers:
    - name: flask-app-container
     image: flask-app-testing
     imagePullPolicy: Never
     ports:
      - containerPort: 5000
       protocol: TCP
Flask_ingress.yaml:
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: flask-app-ingress
 annotations:
 kubernetes.io/ingress.class: nginx
  nginx.ingress.kubernetes.io/ssl-redirect: "false"
spec:
 # ingressClassName: nginx
 rules:
  - http:
    paths:
 backend:
       service:
         name: flask-app-service
         port:
```

number: 5000

path:/

pathType: Prefix

# Flask\_service.yaml:

apiVersion: v1 kind: Service metadata:

name: flask-app-service

spec:

type: ClusterIP

ports:

- port: 5000

selector:

app: flask-app